ICCRTS 2010

Commando DienstenCentra
Ministerie van Defensie

Radboud University Nijmegen

June 2010

Web Based Dynamic Workflows Systems for C2 of Military Operations

Jan Martin Jansen, Bas Lijnse, Rinus Plasmeijer, Tim Grant
Presenter: Ariën van der Wal
Nettherlands Defense Academy
Radboud University Nijmegen

Overview

- Why Workflow tooling for C2 ?
- Workflow tooling
- The iTask System
- iTask applications in the C2 domain
- Analysis of suitability of iTask for C2
- Future Work
- Questions

Why Workflow Tooling for C2?

- Command & Control: networked activity involving many systems and people in a distributed setting
- Complex planning and coordination
- Much information available, but difficult to bring info to the right person at the right moment
- Nowadays: no centralized C2 but distributed decision making
- In asymmetrical warfare Information is the most important weapon

Why Workflow Tooling for C2?

- Recent years
  - Focus (NCW, NEC) has been on information sharing and exchange
- Real problem is coordination and control
  - Getting the right information at the right place at the right moment
  - Getting feedback on actions
  - Maintaining overview on what is going on
  - Adapting actions due to changing circumstances

Workflow Management Systems WFMS

- WFMS
  - Applications that generate, coordinate and monitor tasks performed by human workers using computers
- Examples
  - Claim Handling for Insurance Companies
  - Web shops and Internet banking
  - Enterprise Resource Planning
- Characteristics
  - Tasks can depend on each other and must be performed sequentially
  - Independent tasks can be executed in parallel
  - WFMS coordinates the activities

WFMS for Command and Control?

- WFMS seem to be useful for supporting C2 of military operations

But

- Available Systems
  - rather static
  - cannot adapt easily to changing circumstances
- Focus on flow of control and not of data
  - difficult to parameterise workflows using this data
- WFMS are limited in set of Workflow patterns

iTask: A toolkit for Dynamic Workflows

We need Dynamic Workflow Management Systems!

iTask is made with

iTask: A toolkit for Dynamic Workflows

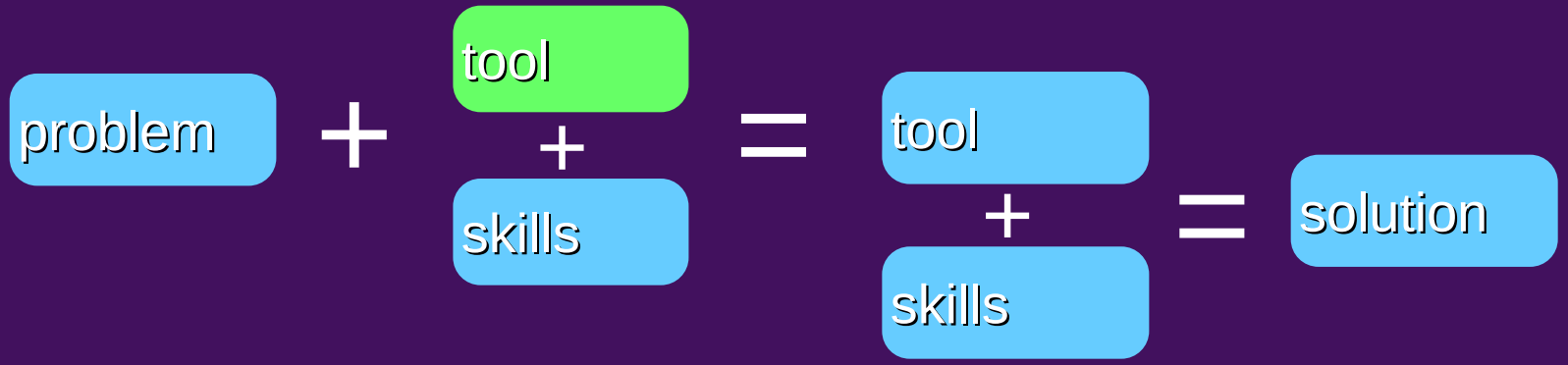
- iTask: library in Functional Programming language Clean for construction of Dynamic Workflow apps
- Developed at Radboud University Nijmegen, the Netherlands
- Clean: start-of-the-art functional programming language and compiler (> 25 years of research) including:
  - Static type checking, Lazy evaluation
  - Generics: Functions that work for all types
  - Dynamics: run-time linking of new code
  - Client side interpreter for executing parts of programs in web-browser

iTask: A toolkit for Dynamic Workflows

toolkit for building
workflow support systems

iTask: A toolkit for Dynamic Workflows

tools to build tools indirection



Clean developer

workflow developer

workflow user

iTask: A toolkit for Dynamic Workflows

Concepts of iTask

- Task: work to be performed by user/computer or both
  - Task can be a single piece of work
  - Task can be a combination of other tasks
- Every task returns a result when it is finished
  - Result can be seen as goal of the task
  - Result can be used for creation of new tasks!
- Tasks can be combined into new tasks by so-called combinators
  - Sequential tasks (with data dependency)
  - Parallel tasks (and, or, conditional)
  - Choice between tasks
  - Task adaptation (exception, change)

iTask: A toolkit for Dynamic Workflows

Properties of iTasks

- iTask applications are distributed client-server apps
  - Web-based user interface
  - No installation of software needed
  - E-mail like interface
- Automatic generation of Web forms from types
- Automatic updates of data with changes in web-forms
- Generic data storage and information exchange with other applications
- Applications generated from single source in Clean
  - No HTML, JavaScript programming, etc needed

Dealing with Dynamic Behaviour

iTask Applications can be dynamic in many ways

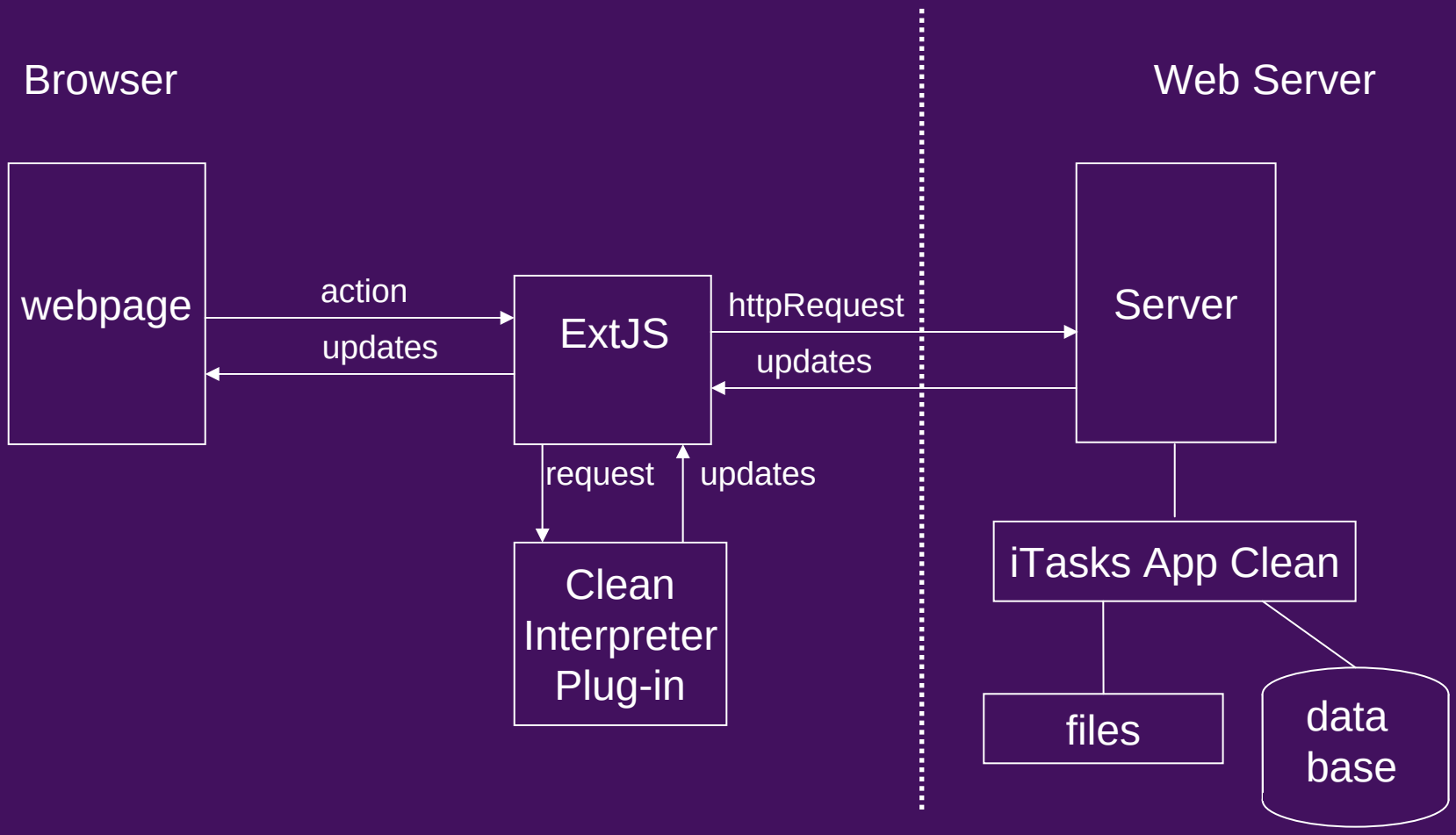
- New actions can depend on outcome of previous actions (data dependency)
- Actions can be stopped and alternative actions can be started (exception)
  - Used to separate (anticipated) uncommon borderline cases from regular workflow
- Action can be replaced ad-hoc by alternative actions (change)
  - Used for unanticipated circumstances

Dealing with Dynamic Behaviour

Examples of changes

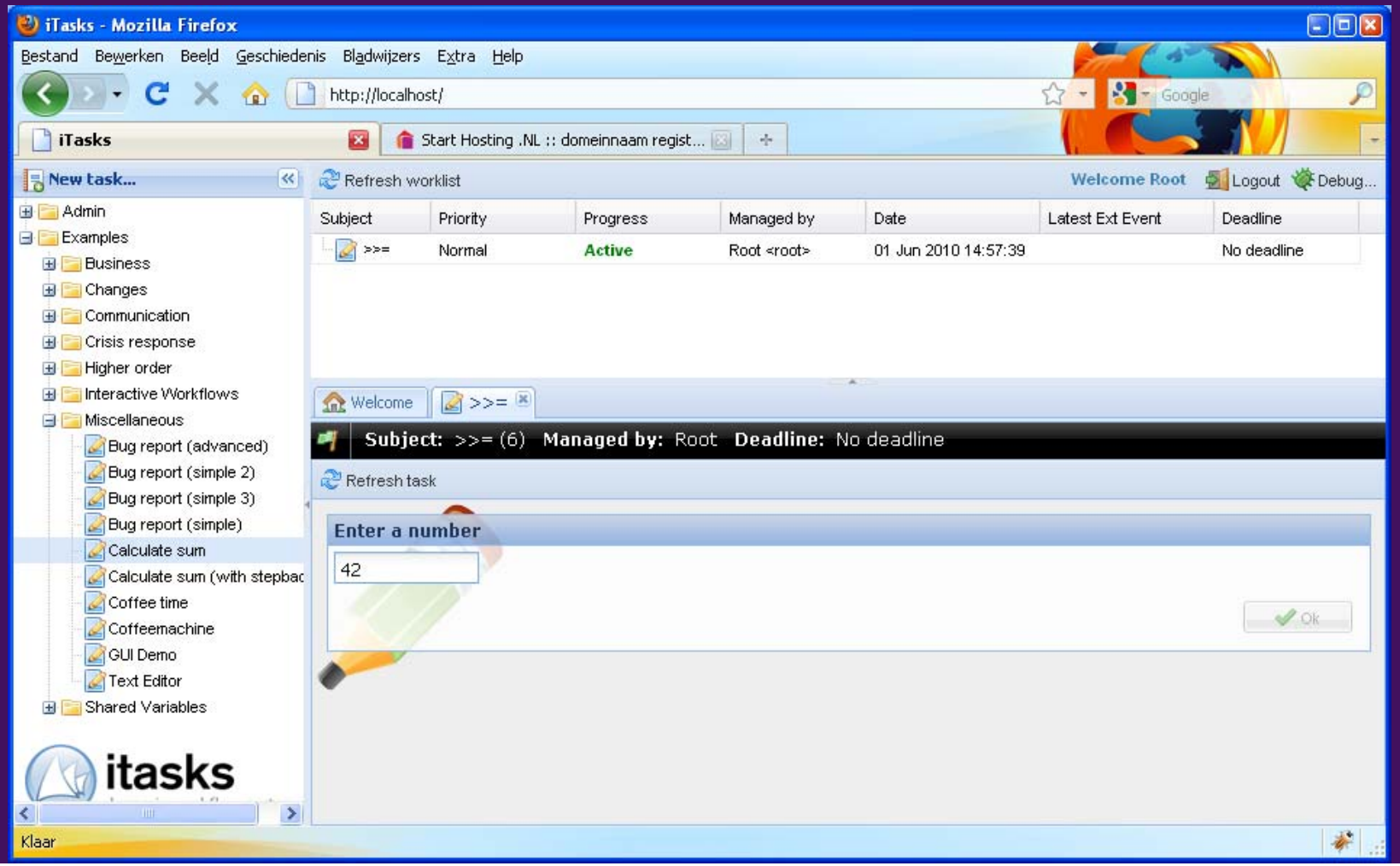
- Re-allocate a task to another user
- Supervise all tasks of a specific user (e.g. trainee) by another user
- Attach a deadline to a task already under execution
- Replace a task (or complete sub-workflow) by ad-hoc entering information into a form (information obtained outside the workflow)
- Replace a form fill-in task by a complex workflow
- Create an ad-hoc workflow interactively

Architecture of iTask Application



Examples of iTasks: Basic Tasks

myTask :: Task Int
myTask = enterInformation “Enter a number"



Examples of iTasks: Basic Tasks

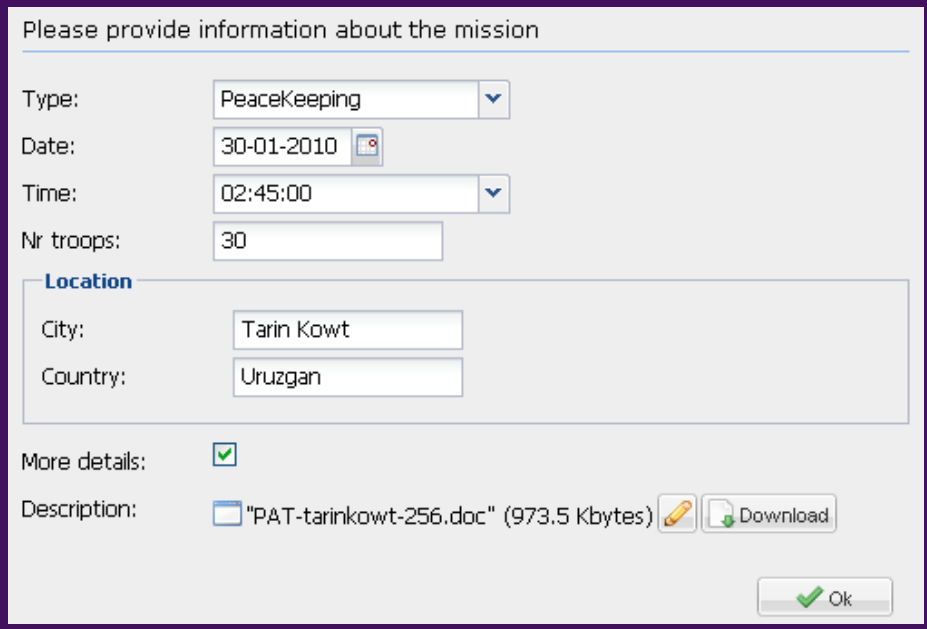
Another editor can be made by just changing the type!

```
enterMission :: Task Mission
enterMission = enterInformation "Please provide information about the mission"

:: Mission =
        { type          :: MissionType
        , date          :: Date
        , time          :: Time
        , nrTroops      :: Int
        , location      :: Location
        , moreDetails   :: Bool
        , description   :: Document
        }

:: MissionType = PeaceKeeping     |
                 CounterTerrorism |
                 SpecialService     | IntelOperation  |  Other String

::Location = {city::String,country::String}
```

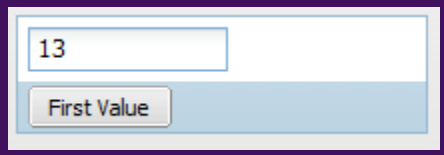


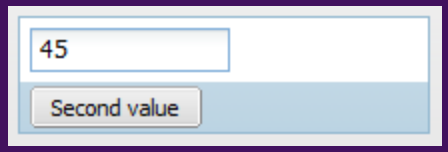
Examples of iTasks: Combinators

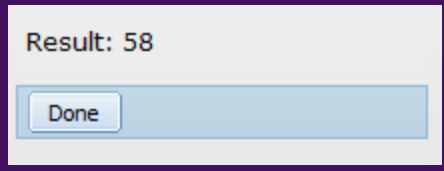
>>= Sequence Combinator,
do tasks after each other and use result

return turn a value into a task

```
addTask :: Task Int
addTask =                    enterInformation “First Value”
          >>= \first      ->  enterInformation “Second
    Value”
          >>= \second ->  return (first + second)
```








```
addVB = addTask >>= showMessageAbout “Result”
```

Other Combinators

Tasks can be assigned to user

user @: task

Tasks can be executed in parallel

Or, And, ad-hoc parallelism

anyTask [$task_1$, … , $task_n$]
allTasks [$task_1$, … , $task_n$]
conditionTask condition [$task_1$, … , $task_n$]

Other Combinators

- Attaching time-out to task
- Reading – Writing info to persistent storage (databases)
- ….

Example: Executing a Mission

```
startMission =
 enterMission >>=
 planActions   >>=
 performMission

enterMission :: Task Mission
enterMission = enterInformation "Please provide information about the mission“

planActions :: Mission -> Task [Action]
planActions mission = ….. // determine the needed actions depending on mission

performMission actions = allTasks actions // execute actions in parallel
```

Why iTask for Military Operations?

- Combination of control and data
  - New tasks can depend on outcome of tasks
- iTask has the right abstraction mechanisms
  - Complex dynamic behaviour can be easily expressed
- Embedded in Programming language
  - Complex algorithms can be used to create tasks
- Can be used for training and simulation
- iTask can be used for formalisation of Standard Operational Procedures

Military Application Areas

- Preparation of Deployment for Military and Peace Keeping Operations
  - complex planning involving many parties
  - activities: logistics, transport, intelligence, C2 and communication, procurement, protection, budget
- Intelligence Operations in Asymmetric Warfare
  - timely gathering of information and bringing this to the right person(s)
- Crisis Management and Cimic (Civil Military Cooperation)
- …..

Evaluation of iTask

- iTask is not C2 or Crisis Response application itself, but a tool to build such applications!
- Difficult to evaluate!
- We tried using general criteria form the literature:
  Suzanne Jul, ISCRAM 2007
  who`s really on first?
  a domain-level user, task and context analysis for response technology

Evaluation of iTask: Results

iTask is strong at:

- Just-in-time Learning
  - Providing Information to people so that they know what to do
  - Applications can often be used without prior training
- Responsive driven tasks
  - Workflow systems coordinate the work to be done
- Cooperation and Collaboration
  - iTask applications coordinate the activities that several people should perform
- Flexibility
  - iTask supports data dependent workflows, exceptions and changes

Evaluation of iTask: Results

iTask should improve at:

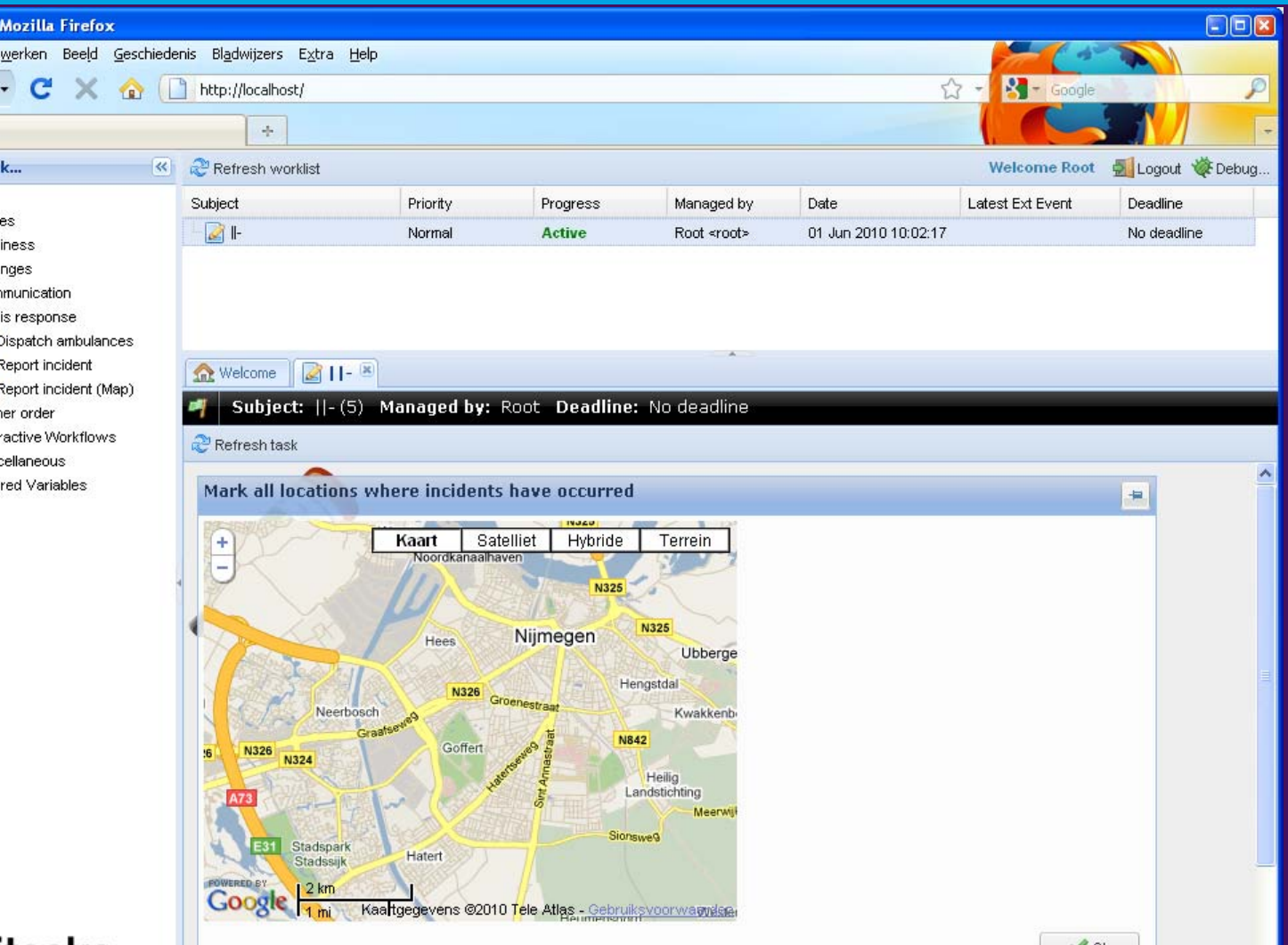
- Supporting users to collaborate on tasks
  - Work together on same task
  - Discuss / Chat about tasks

- Adapting in response to changing circumstances
  - iTask supports changes of tasks, but how should this be offered to an end-user?

Things To Do

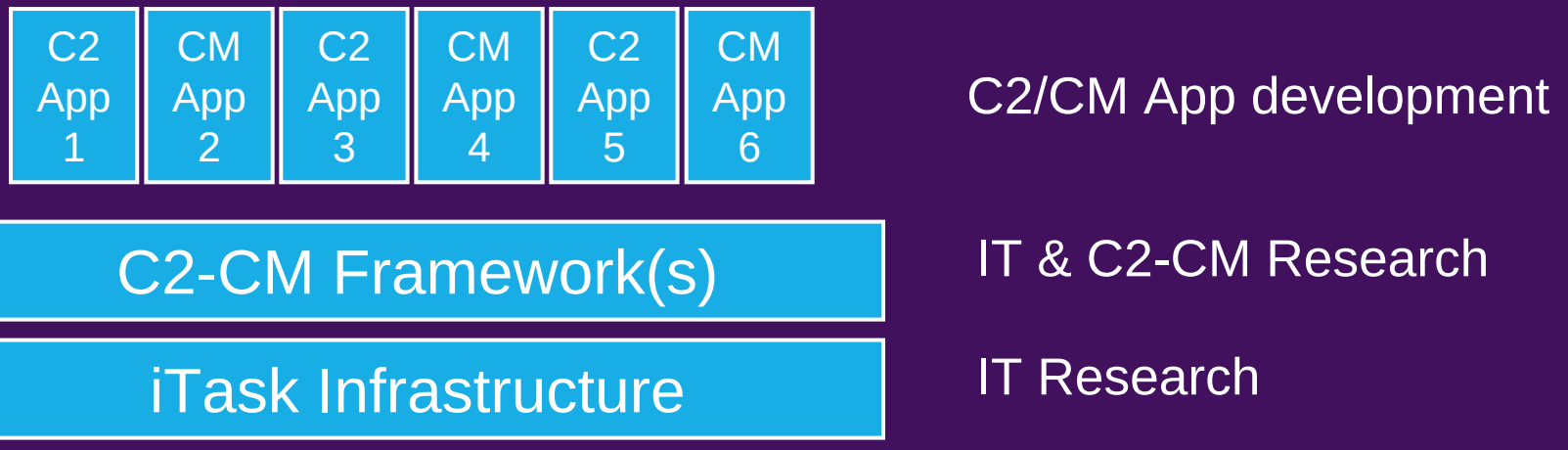
- Better Collaboration
  - Working together on the same task
  - Chatting about tasks
- On-the-fly adaptable Workflows
  - Providing an Interface to monitor tasks and progression
  - Providing a graphical Interface to define workflows interactively
- Integration with other (Web)Tools
  - Web 2.0 like applications: Mash-Ups, GoogleMaps etc
  - Legacy Systems
  - Access to knowledge bases
- Creation of Frameworks
  - Prototype Applications for C2 and CM

ntegration of Web-services



rameworks

- Creation of Generic Framework(s) for a Variety of C2 and CM Operations



se study
dutch coastguard

Questions